The Dissertation Committee for YI LUO

certifies that this is the approved version of the following dissertation:

# An energy-based model of areas tiling the cortex and its fast computation

Committee:

Alexander Huth, Supervisor

Vernita Gordon, Co-Supervisor

# An energy-based model of areas tiling the cortex and its fast computation

by

**YI LUO**

**Thesis**

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

**Master of Arts**

**The University of Texas at Austin**

December 2019

# Acknowledgments

Thank Prof.Alex Huth for his instructions on computational neuroscience, supporting me transition to a new research direction.

Thank Prof.Vernita Gordon for her instructions on biophysics research.

Thank physics department's support.

Thank my parents for always getting my back.

YI LUO

*The University of Texas at Austin*

*December 2019*

# An energy-based model of areas tiling the cortex and its fast computation

YI LUO, M.A.
The University of Texas at Austin, 2019

Supervisor: Alexander Huth

Each human cerebral cortex is a highly folded unique map. In order to combine the surface-based cortex data among subjects, we have developed an energy-based model of areas tiling human cerebral cortex across individuals. This model assumes that the cortical map in each individual subject is a sample from a single underlying probability distribution, allowing for higher localization accuracy of structural and functional features of the human brain. We update parameters using Markov Chain Monte Carlo (MCMC) and Gibbs sampling for this hierarchical Bayesian model. However, this procedure is very computationally costly in practice. To address this issue, in this work we have applied several approximation methods including biharmonic matrix approximation (BHA) and sparse sampling for faster Gibbs sampling. We use time, jumping distance, instability and Kullback–Leibler divergence as our comparison metric for efficiency and accuracy. We find out sparse, close, close-sparse

mixed and BHA are promising approximation methods to make the model practical at different phases.

# Contents

# List of Figures

# Chapter 1

# Introduction

## 1.1 Cerebral cortex

The cerebral cortex is the thin outer layer of neural tissue of the cerebrum of the brain in humans and other mammals. Human cerebral cortex is the largest site of neural integration in the central nervous system where most information processing occurs. Human cerebral cortex has between 14 and 16 billion neurons. The cerebral cortex is separated into the left and right cerebral hemisphere by the longitudinal fissure. The two hemispheres are joined beneath the cortex by the corpus callosum.[4] The cerebral cortex is highly folded, providing a greater surface area. The folds in the brain add to its surface area and therefore increase the amount of gray matter and the quantity of information that can be processed.[5] The cerebral cortex is the most highly developed part of the human brain and is responsible for attention, perception, awareness, thought, memory, language and consciousness. The cerebral cortex contains sensory areas and motor areas. Sensory areas receive input from the thalamus and process information related to the senses while motor areas regulate voluntary movement. It also has specific functions such as sound in the auditory cortex and sight in the visual cortex.

### 1.1.1 Cortical maps

Although the cerebral cortex is a 3-D sheet of several millimeters thick, the 2-D unfolded structure of the cortical is closely linked to its biological function. Cortical maps are 2-D cortical surface organizations that have been identified as performing a specific information processing function (texture maps, color maps, contour maps, etc.).[6] Bruce Fischl stated in his 1999 paper: "(1) many functional dimensions (e.g., retinotopy, orientation tuning, ocular dominance, somatotopy, tonotopy) are mapped on the cortical surface, (2) these mapped parameters vary much more rapidly in the two dimensions parallel to the surface than they do through the several millimeters of cortical thickness, and (3) different cortical areas are arranged in a characteristic pattern, or mosaic, across the cortical surface."[7] This thesis is trying to find the intrinsic unfolded structure of the cortical surface map by mosaic tile arrangement. Figure 1.1 is a demonstration of unfolding a 3-D sheet of the cerebral cortex to a 2-D flat map. The cortical flat map is obtained by conformal mapping of the highly curved cortical surface to a 2-D plane.[1]

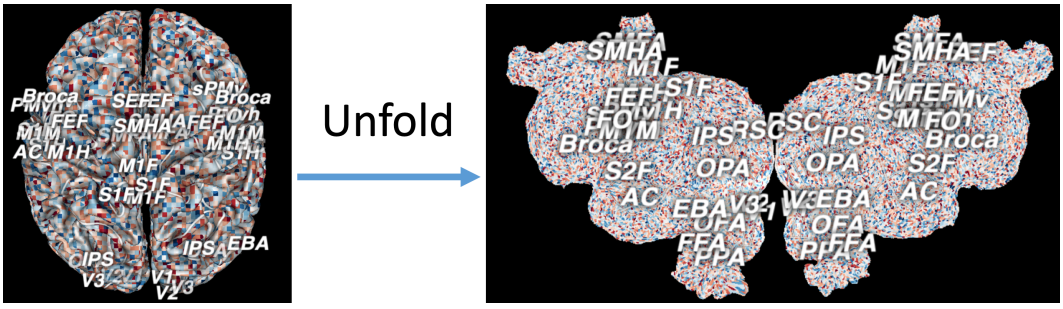


Figure 1.1: A demonstration of unfolding a 3-D sheet of the cerebral cortex (left) to a 2-D flat map (right), drawn with Pycortex (a python-based toolkit for surface visualization)[1].

### 1.1.2 Registration among subjects

Everyone has a unique cortical map. The question rises up when we study more than one subject: how do we determine which response in one subject corresponds to a given response in another subject? Alex Huth summarised three existing answers to this question in his 2015 paper [3]:

- "One answer is to use anatomy. Many existing methods align anatomical images of brains from multiple subjects in either volumetric space or surface space. These methods assume that brain anatomy and function are perfectly correlated, and that any deviation from this relationship is noise. Yet this assumption has repeatedly been proven false."[3]

- "Another way to find correspondence between subjects is by using functional responses. Some new methods find corresponding temporal patterns of functional activation in each subject given responses to a complex natural stimulus, such as a movie. These methods have proven much more successful at predicting functional data in new subjects than anatomical methods and may in fact be near-optimal for mapping data from one subject to another. Working purely in functional space makes it possible to disregard anatomical differences between subjects, but it also means that these methods provide little information about the organization of cortical maps. Indeed they would work just as well even if cortical maps were completely different in each subject. We might be able to learn more about how functional representations on the cortex are organized if we can take both function and anatomy into account."[3]

- "A third way to find correspondence between subjects is the region-of-interest (ROI) analysis, which combines both anatomical and functional constraints. In this method, an ROI is defined in each subject using a functional localizer contrast. Among voxels that respond significantly to the localizer a single

cluster is selected as the ROI based on its approximate anatomical location. Then on a different dataset, average functional responses within that ROI are compared across subjects. This method does not assume exact anatomical correspondence across subjects. However, for many areas of the brain there is no known localizer, making it impossible to apply the ROI method."[3]

So we need a new method to register cortical maps among subjects that reflects both anatomical and functional features.

## 1.2 Energy-Based Models

Statistical modeling is a simplified, mathematically-formalized way to approximate reality and to make predictions. Statistical models contain independent and dependent variables. The main purpose of modeling is to find out dependencies between variables. By capturing those dependencies, a model can be used to answer questions about the values of unknown variables given the values of known variables. Energy-Based Models (EBMs) capture dependencies between variables by associating a scalar energy to each configuration of the variables.

### 1.2.1 The framework

Energy-Based Models' framework has two major steps: inference and learning.

- Inference builds a function that maps each point of an input space to a single scalar, which is called "energy".
- Learning is a data-driven process that shapes the energy in such a way that the desired configurations get assigned low energies, while the incorrect ones are given high energies. During the leanring processe, loss function, minimized during learning, is used to measure the quality of the available energy functions.

Within this framework, the wide choice of energy functions and loss functions allows for the design of many types of statistical models, both probabilistic and non-probabilistic. Energy-based learning can be seen as an alternative to probabilistic estimation for prediction, classification, or decision-making tasks. Because there is no requirement for proper normalization, energy-based approaches avoid the problems associated with estimating the normalization constant in probabilistic models. Furthermore, the absence of the normalization condition allows for much more flexibility in the design of learning machines. The EBM approach provides a common theoretical framework for many learning models, including traditional discriminative and generative approaches, as well as graph-transformer networks, conditional random fields, maximum margin Markov networks, and several manifold learning methods.[8, 9]

### 1.2.2 Probabilistic EBM

A probabilistic model is based on the theory of probability where randomness plays a role in predicting future events. Many probabilistic models can be viewed as special types of energy-based models in which the energy function satisfies certain normalizability conditions, and in which the loss function, optimized by learning, has a particular form. The simplest and most common method for constructing a energy-based probabilistic model is to turning a collection of arbitrary energies into a collection of numbers between 0 and 1 whose sum (or integral) is 1. This can be achieved through the Gibbs distribution (also called Boltzmann distribution)[10]:

$$P_i = \frac{e^{-\beta E_i}}{\sum_i e^{-\beta E_i}} \tag{1.1}$$

where $\beta$ is the inverse temperature, $E_i$ the energy of state $i$ and $P_i$ the probability of the system in state $i$. The denominator is called the partition function in statistical physics. The choice of the Gibbs distribution may seem arbitrary, but

other probability distributions can be obtained (or approximated) through a suitable redefinition of the energy function. Whether the numbers obtained this way are good probability estimates does not depend on how energies are turned into probabilities, but on how energy $E$ is estimated from data. It should be noted that the above transformation of energies into probabilities is only possible if the partition function converges. This somewhat restricts the energy functions and sampling candidates.[8, 9, 11]

## 1.3 PrAGMATiC

Knowing the power of the probabilistic energy-based model and facing the cortical registration problems, Alex Huth, my advisor, created a probabilistic energy-based model that can register cortical maps among subjects. He established a mapping that specifies a unique correspondence between each location in one brain and the corresponding location in another to relate and compare anatomical or functional features across subjects. [3]

### 1.3.1 The model

PrAGMATiC, a probabilistic and generative model of areas tiling the cortex, is a hierarchical Bayesian generative model of cortical maps. This model assumes that the cortical map in each individual subject is a sample from a single underlying probability distribution. Learning the parameters of this distribution reveals the properties of a cortical map that are common across a group of subjects. This model consists of two parts: an arrangement model and an emission model. The arrangement model uses a centroid-spring network to describe cortical topography. This flexibility brought by the springs can account for substantial individual differences in the shape, size, and anatomical location of functional brain areas. The emission model then generates predicted functional cortical maps based on the arrangement of the

centroids.[3, 12]

### 1.3.2 The challenge

The arrangement model of PrAGMATiC contains a Gibbs sampler. In the sampler, we need to find the total spring energy for every possible location on the entire cortex, for every centroid, on every sweep. The total spring energy depends on geodesic distance matrix which has to be updated after every time a new centroid is sampled. The recursive sweeping over the whole cortical map is extremely expensive both in time and space complexity which makes the model impractical. The cortical map we study has 150k units in the visible layer, which gives out a huge geodesic distance matrix in the size of (150k×150k). It is impossible to store or process full distance geodesic matrix. Without proper approximation, this model can be prohibitively expensive to fit.[3]

### 1.3.3 Statement of purpose

In this thesis, I will explain the mathematical formulation of this PrAGMATiC model and show how the various parameters can be learned. Based on Alex's work and the challenge PrAGMATiC is facing, I propose to develop several approximation methods for the arrangement model to make it practical to use. I propose the best approximation methods by comparing time and space complexity and approximation accuracy among them.

# Chapter 2

# Build a spring system model

## 2.1 Modeling tools

I will introduce some physical and mathematical tools that will be used in the spring system model of tiling the cortical map.

### 2.1.1 A spring system

A spring system or spring network is a physical model of centers of mass connected to each other with a spring or springs of given stiffness and lengths. It extends Hooke's law to higher dimensions. With linear springs and small deformation a spring system can be cast as a system of linear equations or equivalently as an energy minimization problem. Assuming $k_i$ is the spring constant, $x_i$ the length, $x_{i0}$ the free length for the $i$th spring, the total energy $E$ can be expressed by summing over all springs:

$$E = \sum_i \frac{1}{2} k_i (x_i - x_{i0})^2 \tag{2.1}$$

### 2.1.2 Bayes theorem

Bayes' theorem is a simple mathematical formula used for calculating conditional probabilities. Bayesian theorem derives the posterior probability from a prior probability and a likelihood function which is stated mathematically as the following:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{2.2}$$

where $A$ is the hypothesis, whose probability may be affected by evidence. Usually there are competing hypotheses, and the task is to determine which is the most probable. $P(A)$, the prior probability, is the estimate of the probability of the hypothesis $A$ independently before the data $B$, the current evidence, is observed. $B$, the evidence, corresponds to new data that were not used in computing the prior probability. $P(B)$ is the estimate of the probability of the evidence $B$ independently. $P(B|A)$ is the probability of observing $B$ given $A$, and is called the likelihood. As a function of $B$ with $A$ fixed, it indicates the compatibility of the evidence with the given hypothesis. $P(A|B)$, the posterior probability, is the probability of $A$ given $B$. This is what we are looking for: the probability of a hypothesis given the observed evidence.[13] Bayesian models use Bayes' theorem to compute conditional probabilities after obtaining new data.

### 2.1.3 Markov chain Monte Carlo

Markov chain is a sequence of possible events in which the probability of each event depends only on the state of the previous event. Its future and past states are independent given the current state. A example of a Markov chain is a simple random walk on integer number line, $Z$, which starts at 0 and at each step moves +1 or -1 with equal probability. Monte Carlo methods use random numbers to conduct "mathematical experiments". Markov chain Monte Carlo methods create samples

from a possibly multi-dimensional continuous random variable, with probability density proportional to a known function. Suppose we have a probability distribution $f$ on a set $S$ and our goal is to generate random elements of $S$ with distribution $f$. We can achieve this by inventing a Markov chain whose equilibrium distribution is $f$ and then simulate the chain for a long time. The more steps that are included, the longer the chain is, the more closely the distribution of the sample matches the desired distribution $f$. [2, 14–16]

### 2.1.4 Gibbs sampling

Gibbs sampling or a Gibbs sampler is an approach to design Markov chains with a given equilibrium distribution. It is well suited for problems containing many random elements such that the conditional distribution of each element is known, especially when direct sampling is difficult. The idea in Gibbs sampling is to generate samples by sweeping through each variable to sample from its conditional distribution with the remaining variables fixed to their current values. Figure 2.1 demonstrates the pseudo algorithm of one iteration for Gibbs sampling. Within one iteration(sweep), we sample and update the variables one by one while fixing other variables fixed. We collect the sampled results $x_1^{(i)}, x_2^{(i)}, ..., x_D^{(i)}$ as our sampling distribution of the $i$th iteration from the sampler. Generally, samples from the beginning of the chain (the burn-in period) may not accurately represent the desired distribution and are usually discarded. [2, 14, 15]

Initialize $x^{(0)} \sim q(x)$
**for** iteration $i = 1, 2, \ldots$ **do**
$x_1^{(i)} \sim p(X_1 = x_1 | X_2 = x_2^{(i-1)}, X_3 = x_3^{(i-1)}, \ldots, X_D = x_D^{(i-1)})$
$x_2^{(i)} \sim p(X_2 = x_2 | X_1 = x_1^{(i)}, X_3 = x_3^{(i-1)}, \ldots, X_D = x_D^{(i-1)})$
$\vdots$
$x_D^{(i)} \sim p(X_D = x_D | X_1 = x_1^{(i)}, X_2 = x_2^{(i)}, \ldots, X_D = x_{D-1}^{(i)})$
**end for**

Figure 2.1: Algorithm for Gibbs sampling, reproduction from Ilker's paper(2012)[2]. Within one iteration, we sample and update the variables one by one while keeping other variables fixed.

### 2.1.5 Maximum likelihood estimation(MLE)

Maximum likelihood estimation (MLE) is a method of estimating the parameters of a probability distribution by maximizing a likelihood function, so that under the assumed statistical model the observed data is most probable. The point in the parameter space that maximizes the likelihood function is called the maximum likelihood estimate. In Equation 2.2, the likelihood, $P(B|A)$, is proportional to the posterior probability, $P(A|B)$. MLE is a special case of maximum a posterior estimation that assumes a uniform prior distribution of the parameters. [17]

### 2.1.6 Voronoi diagram

Assuming centroids are points on a surface, a Voronoi diagram is a way of tiling the surface into regions close to each of a given set of centroids. For each centroid there is a corresponding tile area consisting of all points of the plane closer to that centroid than to any other. These tiles are called Voronoi cells. The Voronoi diagram can also be achieved by Delaunay triangulation.

## 2.2 Building the model

Now we apply the tools above to build the PrAGMATiC model. The model for each subject is instantiated as a two-layer Bayesian network, with one hidden layer and one visible layer. In the hidden layer, we use a spring system to represent cortical topography, which can be considered as an arrangement model. In the visible layer, every point is assigned an observed functional value, which can be considered as an emission model. The arrangement model tiles the cortical surface and the emission model tries to generate observed brain data given a specific arrangement. Both sub-models are probabilistic EBMs. Applying Bayes' theorem we get the likelihood by combing two sub-models. Parameters can be learned using maximum likelihood estimation. The model is fit separately for each cortical hemisphere. Due to the limitation of time and storage, without loss of generality, we only study the left hemisphere in this thesis.

### 2.2.1 Arrangement model

In the arrangement model, we suppose that the cortex of each subject is tiled with convex areas. The location of each tile area is determined by the location of its centroid, which is a single point on the cortical surface. There are some known cortical landmarks, which are identified separately in each subject. Centroids and landmarks function as centers of mass, connecting each other by springs. Centroid locations depend on the locations of neighboring centroids and landmarks and the springs connecting them. The hidden layer units are the locations of the tile area centroids. The location of centroid $i$ in subject $s$ is called $h_{is}$, and the collection of all hidden units in subject $s$ is called $H_s$. Given lengths and stiffness of springs, the arrangement probability can be expressed with Equation 1.1 and Equation 2.1:[3]

$$P(H; L, K) = \frac{e^{-\beta E(H;L,K)}}{\sum e^{-\beta E(H;L,K)}} \tag{2.3}$$

$$E(H; L, K) = \frac{1}{4} \sum_{i,j,s} K_{i,j} (\mathcal{D}_{i,j,s}(H) - L_{i,j})^2 \tag{2.4}$$

where $K_{i,j}$ is the spring constant for the spring linking centroids $i$ and $j$, $L_{i,j}$ is the ideal length of the spring linking $i$ and $j$, and $\mathcal{D}_{i,j,s}(H)$ is the geodesic distance across the cortex between centroids $i$ and $j$ in subject $s$ and in the arrangement $H$. This energy function, $E(H; L, K)$, is exactly the sum over the spring potential energy for all spring connections in the model. This model is shown schematically in Figure 2.2.[3]

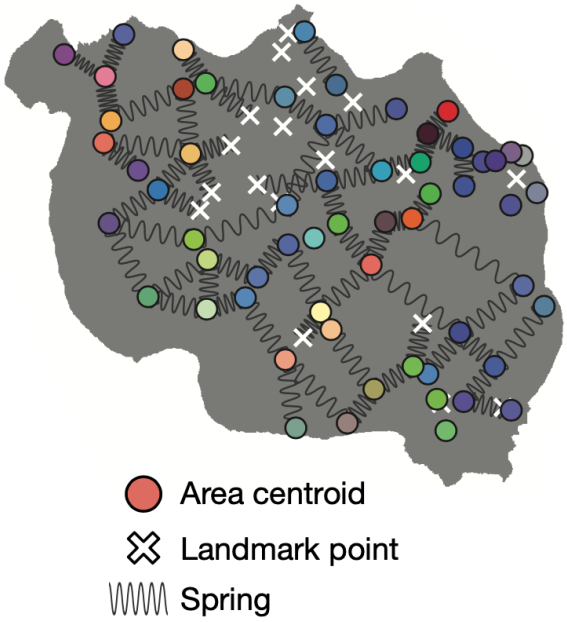


Figure 2.2: Diagram of the arrangement model, reproduction from Alex's paper (2015)[3]. Colored circles are centroids; crosses are identified cortical landmarks. Both centroids and landmarks function as centers of mass, connecting each other by springs, forming a spring system on the hidden layer of the cortex model.

The hyperparameter $\beta$ is the inverse temperature of the spring system introduced by us to control the model behavior. If $\beta$ is large, then the system has low temperature, and the springs will not deviate much from their ideal lengths; if $\beta$ is small, then the system has high temperature, and the springs will deviate a lot from

their ideal lengths.

In the Gibbs sampler we update the location of each centroid, $h_{is}$, based on the locations of the other centroids, $H_{\setminus is}$. This is done by finding the total spring energy given each possible location of the selected centroid, summing the non-normalized probabilities to find the normalizing constant, and then sampling from the resulting multinomial distribution. When sampling one possible location $h_{is} = l^*$, Equation 2.4 and Equation 2.3 become:

$$E(h_{i,s} = l) = \frac{1}{4} \sum_{i,j,s} K_j (\mathcal{D}_{i,j,s}(h_{i,s} = l, H_{\setminus is}) - L_{i,j})^2 \tag{2.5}$$

$$P(h_{i,s} = l^*, H_{\setminus is}) = \frac{e^{-\beta E(h_{i,s}=l^*)}}{\sum_l e^{-\beta E(h_{i,s}=l^*)}} \tag{2.6}$$

Figure 2.3 shows the conditional probability map for the location of each centroid, $h_{is}$, given other centroids, $H_{\setminus is}$, fixed. Notice that for a point candidate $l^*$ to the centroid, the lower spring energy $E$ is, the larger $e^{-\beta E}$ will be, and the larger probability the centroid will be sampled at this point. So it is equivalent to say, the sampler is looking for the configuration of springs that has a minimum energy.

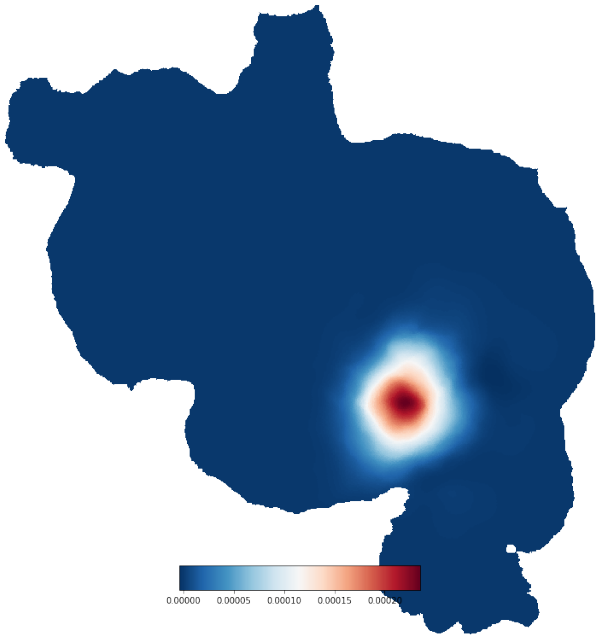


Figure 2.3: Example of probability map of $P(h_{is})$ for $i$th centroid in subject $s$. The red area has smaller energy and therefore larger probability the $i$th centroid will be sampled at.

### 2.2.2 Emission model

In the emission model, each vertex is associated with an observed functional value which can be represented by a vector or a scalar. We call the cortical map here as visible layer. The visible layer units are vertices on the cortical surface mesh. Each vertex point is associated with an observed functional value which can be a scalar or a vector. The observed value for visible unit $l$ in subject $s$ is called $V_{ls}$, and the collection of all visible units in a subject is called $V_s$. We assume that all visible units are independent of each other given the hidden layer units. The cortical surface mesh is tile by the locations of centroids via Delaunay triangulation. Each tile is a Voronoi cell. The mean functional value for the $i$th tile area is denoted $M_i$. We assume all points within this tile area have the functional values as samples from a multivariate

Gaussian distribution with mean values, $M_i$, and variance $\sigma_{V,i}$. Emission model then generates predicted functional cortical maps based on the arrangement of tiles.[3] Here we define the squared error of observed functional values as our pseudo energy to build a probabilistic energy based model. Similarly, the emission probability is constructed with Equation 1.1:

$$P(V|H;M) = \frac{e^{-\sigma_V^{-2}E(V|H;M)}}{\sum e^{-\sigma_V^{-2}E(V|H;M)}} \tag{2.7}$$

$$E(V|H;M) = \sum_{l,s}(M_{c(H,l,s)} - V_{l,s})^2 \tag{2.8}$$

Where $V_{l,s}$ is the functional value for visible unit $l$ in subject $s$, and $M_{c(H,l,s)}$ is the mean functional value for the closest hidden layer unit (by geodesic distance across the cortical surface) in the arrangement $H$ to visible layer unit $l$ in subject $s$. The constant $\sigma_V$ is the standard deviation of the Gaussian, which is assumed to be equal in all dimensions. Notice that the $\sigma_V^{-2}$ plays the role of $\beta$ to make the exponent is dimensionless. This model is shown in Figure 2.4.[3]

### 2.2.3 Maximum likelihood estimation

We can get the total probability by joining the arrangement probability from Equation 2.3 and the emission probability from Equation 2.7:

$$P(V,H;M,L,K) = P(H;L;K) \cdot P(V|H;M) \tag{2.9}$$

We use maximum likelihood estimation (MLE) to learn the spring lengths, $L$, the spring constants, $K$, and the area functional means, $M$, based on observed visible unit data, $V$. Maximum likelihood estimation (MLE) requires us to find the gradients. One way to compute gradients we need to integrate over $P(H)$, but that is hard. Assuming we have $N_v$ = 150k vertices and $N_c$ = 128 centroids,

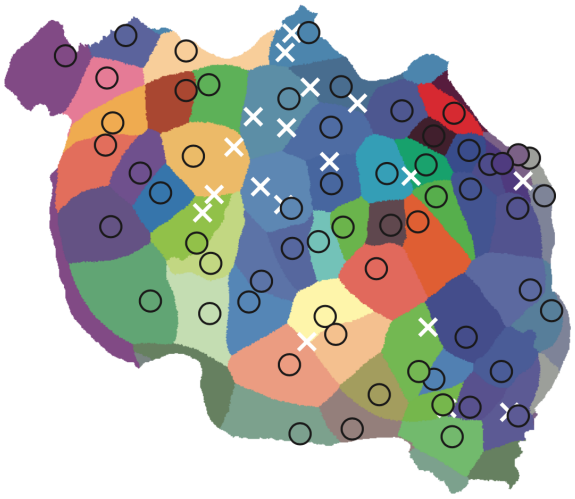

Figure 2.4: Diagram of the emission model, reproduction from Alex's paper (2015)[3]. Black circles are centroids. The color of each cortex tile represents that every points with this color belongs to this tile and their mean functional value which is assigned to the centroid it contains. With arrangement in the hidden layer, the visible layer becomes Mosaic tiles of functional values in the visible layer.

the dimension of $P(H)$ will be $\frac{N_v!}{(N_v-N_c)!} \approx 8.5 \times 10^{446}$ which is beyond computer's computing capacity. So we approximate that integral by drawing $J$ samples from $P(H)$ (this is called Monte Carlo integration). To do that we maintain $J$ parallel Markov chains for each of the $N$ subjects. The average log likelihood is:

$$\mathcal{L} = \frac{1}{N} \sum_s \log P(V_s, H; M, L, K) \tag{2.10}$$

where $N$ is the number of subjects, $s$ is an index across subjects. At each learning step we perform one Gibbs sweep through each of the Markov chains. That is, at step $t$ in chain $j$ and subject $s$ we draw the sample: For each of the $J$ samples we compute the energy gradients for $L, K$ and $M$, as well as the likelihood of the observed data $P(V_s)$. Then we compute the average gradients (for $L$ and $K$) and the weighted average gradient according to the data likelihoods. Finally we update

L, K, and M by taking a small step down these gradients:[3]

$$L^{t+1} = L^t - \epsilon_L \frac{\partial \mathcal{L}(L^t; V)}{\partial L^t} \tag{2.11}$$

$$K^{t+1} = K^t - \epsilon_K \frac{\partial \mathcal{L}(K^t; V)}{\partial K^t} \tag{2.12}$$

$$M^{t+1} = M^t - \epsilon_M \frac{\partial \mathcal{L}(M^t; V)}{\partial M^t} \tag{2.13}$$

## 2.3 Modeling difficulty

### 2.3.1 Hyperparameters

The inverse spring temperature, $\beta$, determines how stiff or loose the springs are. If $\beta$ is very high, the temperature will be low, then the springs will be very stiff, samples of the arrangement $H$ will be highly correlated across iterations. There is strong bias in favor of states with low energy and the model can stuck in a local minimum. If $\beta$ is very low, the temperature will be high, then the springs will be very floppy, samples of the arrangement $H$ will be highly random. The bias will not be so favorable and the quality of the gradient steps will also suffer although the system reaches equilibrium faster. A good solution to beat the trade-off is to start at high temperature and gradually reduce it. We start from a relative high temperature so the hidden layer has enough flexibility to account for substantial individual differences. Then we slowly cool down the system by increasing $\beta$ exponentially ranging from $1.0 \times 10^{-5}$ to $8.1 \times 10^{-2}$ to finalize the learning parameters.[11]

If the standard deviation of the visible unit Gaussian distribution, $\sigma_V$ is very low, then one sample from the arrangement $H$ will always yield much higher likelihood of $V$ than the others, and the weighted average of the gradients across samples will become the difference between the best sample and the other samples.

If $\sigma_V$ is very high, then the likelihood of $V$ will be very similar for all samples, and the weighted average of the gradients will be almost identical to the simple average across gradients, making learning slow. These hyperparameters interact with each other and with the number of tiles as well.[3] We also introduce the entropy which can be calculated from of Equation 2.5's probability to control the modeling stability.

### 2.3.2 Arrangement sampling

The computational complexity of the parameter estimation comes mostly from two operations: drawing samples from the distribution over arrangements $P(H)$ using Gibbs sampling, and computing the likelihood of the observed visible unit data $P(V|H)$. Of these two steps sampling $P(H)$ is much more expensive. In the Gibbs sampling we update the location of each centroid based on the locations of the other centroids. Refer to Equation 2.8 and Equation 2.7, the Gibbs sampling is done by finding the total spring energy given each possible location of the selected centroid, summing these energies to find the normalizing constant. That is, for each possible location, we compute the total spring energy. To compute this distribution exactly we would need to find the total spring energy for every possible location $l^*$ in the entire cortex, for every centroid, on every sweep of the Gibbs sampler. This operation is orders of magnitude more expensive than any other step in the model.[3]

### 2.3.3 Geodesic distance

The geodesic distance is distance between two vertices in a shortest path connecting them on the cortical surface mesh. The geodesic distance between the point $p_i$ and point $p_j$ is labeled as $\mathcal{D}(p_i, p_j)$. For a $n$-point surface mesh, $\mathcal{O}(n^2)$ space is required to store the distance matrix, and, depending on the type of data and distance metric, $\mathcal{O}(n^2 \log n)$ even as much as $\mathcal{O}(n^3)$ time can be required to compute it. The

cortical map we study has 150k units in the visible layer, which result in the geodesic distance matrix size (150k×150k) taking 180 G bytes for memory and it's too large to be precached. This huge matrix has to be fully recalculated at every sweep if not precached.[18]

Accurate geodesic distances calculation is every expensive and impractical. We compute these geodesic distances using a heat-based approximation to the exact geodesic distance across the cortical surface mesh. The heat method is robust, efficient, and simple to implement since it is based on solving a pair of standard linear elliptic problems.[19] The method requires only standard heat diffusion differential equation and can hence be easily applied on the cortical surface mesh. It is accurate enough but not fast enough. In this thesis, we use the result from heat-based approximation as our Ground Truth and keep exploring faster approximation methods.

# Chapter 3

# Approximation methods

## 3.1 Sparse approximation

The intuition of achieving faster computation and less storage is to only use a subset of the points in the cortical map, which is to use sparse network approximation. By homogeneously decreasing the number of cortex points that we allow the centroids to occupy, we created a sparse cortical map that has lower resolution compared with the full one. In this way, we can reduce both the computational and memory demands. The cortical surfaces that we use have approximately 150,000 vertices in left hemisphere. We choose 15,000 random vertices in the left hemisphere to serve as possible centroid locations. This immediately reduced both the computational and memory demands of precaching one row of the geodesic distances matrix by a factor of 10. Most importantly, the full geodesic distance matrix size reduced from 180 G (150k×150k×8 bytes) to 18 G which can be precalculated and precached in the memory. In this case, when we update $H$ in our Gibbs sampler, we can directly draw the geodesic distance we need $\mathcal{D}_{i,j,s}(h_{i,s} = l, H_{\setminus is})$ from the precached geodesic distance matrix without recalculating. This saves time at the cost of accuracy. They key to the trade off here is the sparsity gap, $g_s$. In the case above, we choose the

sparsity gap as 10, which means we take a point from every 10 points as part of our subset. If we raise the sparsity gap, we get less complexity and also less accuracy. If we decrease the sparsity gap to 1 which is the limitation, it is the same as original model. We are handling a 2-D mesh; the Figure 3.1 is a one dimensional demonstration for the sparse approximation.

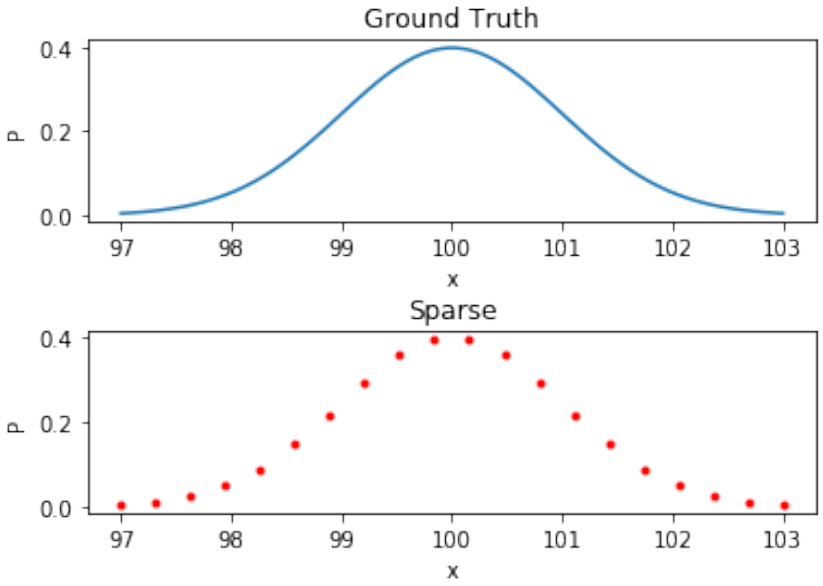


Figure 3.1: One dimensional comparison between ground truth and sparse approximation. x axis is a sample location and y axis is the probability. Sparse approximation takes a subset of ground truth as new dataset for candidate centroid.

## 3.2 Close area approximation

Another way to only use a subset of the cortical map is the close area approximation. As we know from Figure 2.3 the conditional probability map for sampling one centroid is similar to a multi-normal distribution on the mesh. The area near the previous centroid has higher probability for the centroid sampling than points (area) far away. Especially when temperature is low and $\beta$ is high, the location change of the centroid is relative small and local. We propose to make the sampler

short sighted and only takes the close area's dataset as a subset. The sampler can only see a previous centroid's close area which is cut off by a threshold close radius $r_c$. we use this small subset of the points as the candidate for centroids. Figure 3.2 is a one dimensional demonstration for the close area approximation. Then here comes the question: How close should it be? If we have smaller close radius, we have less complexity and less accuracy as well. To make sure the close threshold is safe without losing accuracy we refer to the jumping distance. Jumping distance, $JD$, is the geodesic distance between the location of centroid $i$, $C_i$, at step $t$ and the location of the same centroid at step $t+1$.

$$JD(C_i, t) = \mathcal{D}(C_i^t, C_i^{t+1}) \tag{3.1}$$

Jumping distance can be calculated in each method. We choose the largest jumping distance from the close area approximation to calculate threshold close radius $r_c$:

$$r_c^{t+1} = \max\{JD(C_i, t)\} \times \alpha \tag{3.2}$$

Where $\alpha$ is the trade off scalar factor. When $\alpha$ is smaller, the computation will be faster but less accurate. We fix $\alpha$ at 1 in our simulation. $r_c$ is updated at every iteration. This is extra price we have to pay for the close approximation.

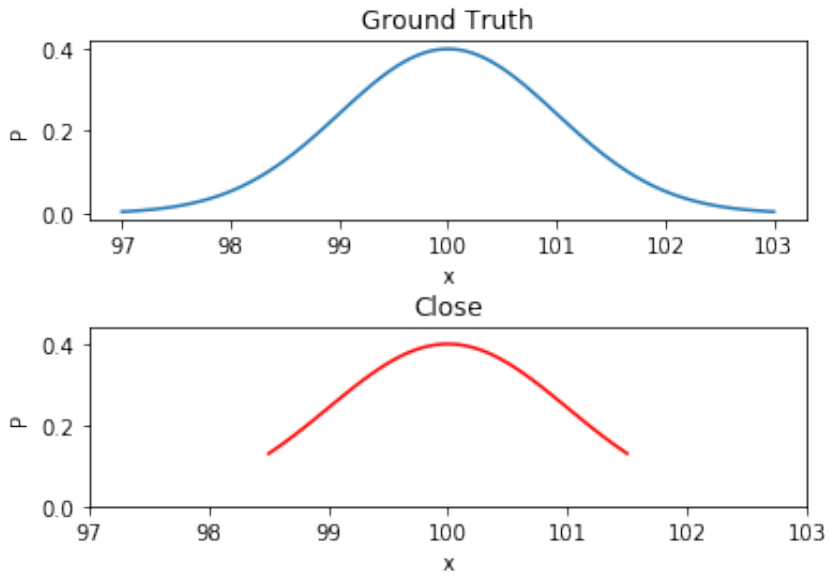


Figure 3.2: One dimensional comparison between ground truth and close area approximation. x axis is a sample location and y axis is the probability. Close area approximation takes a subset of near the peak as new dataset for candidate centroid.

## 3.3 Close and sparse approximation

Sparse and close area approximation are two different methods. It is natural to go on one further step to combine both as a new approximation method. We first take the close area subset of the surface mesh by cutting off at the threshold close radius $r_c$; and then take the sparse subset of the close area subset. In this way we are taking advantages from both models. The two parameters sparsity gap, $g_s$, and close radius, $r_c$, regulate the approximation behavior together. The extra perks of the mixed model is that jumping distance can be directly pulled out from the precached geodesic distance matrix. It will be easy and direct to get $r_c$ by finding the maximum of the jumping distance. The Figure 3.3 is a one dimensional demonstration for the close and sparse approximation.

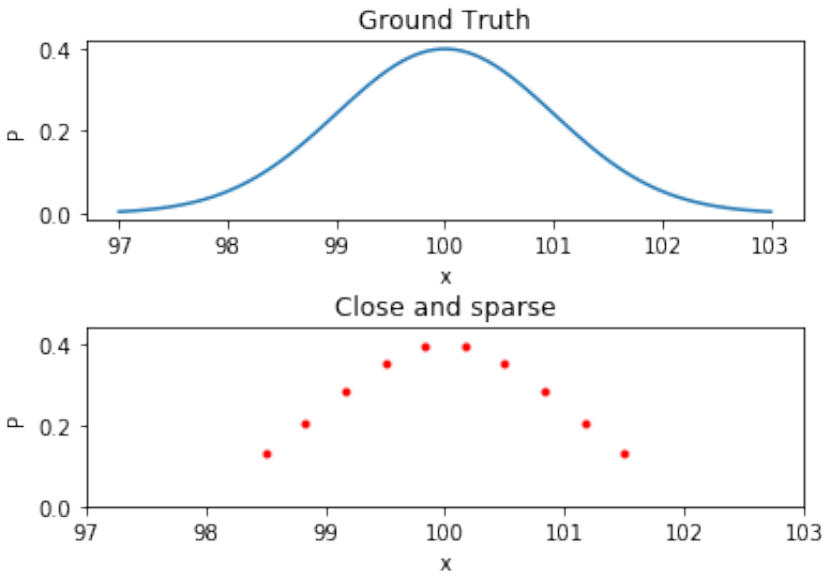


Figure 3.3: One dimensional comparison between ground truth and close and sparse approximation. x axis is a sample location and y axis is the probability. Close and sparse approximation takes a subset of some data near the peak as new dataset for candidate centroid.

## 3.4 Biharmonic matrix approximation (BHA)

Interpolation is a type of estimation, a method of constructing new data points within the range of a discrete set of known data points. When the interpolation comes to 2-D surface mesh, people have developed many powerful tools. One of them is Biharmonic matrix approximation (BHA). BHA is a method for efficiently representing geodesic distance matrices using biharmonic interpolation. These representations can be used to do multidimensional scaling on geodesic distance matrices by the biharmonic method. We define $g$ as a real-valued function on a smooth surface mesh $\mathcal{M}$. Now suppose that we are given $g(b_i)$ for a known set of $l$ points, $\{b_i\}_{i=1}^{l} \in \mathcal{M}$, and wish to interpolate $g(u_i)$ at a different set of $m$ points, $\{u_i\}_{i=1}^{m} \in \mathcal{M}$. The surface mesh $\mathcal{M}$ have $l + m = n$ points in total. One solution

to interpolate $g(u)$ is biharmonic interpolation, which finds a smooth function (i.e. one with continuous second derivative) that passes exactly through $g(b_i)$ for all $i$. Biharmonic interpolation is accomplished by finding a solution to the biharmonic equation subject to Dirichlet boundary conditions given by $g(b)$: [18]

$$\Delta^2 g(u) = 0 \tag{3.3}$$

where $\Delta$ is the Laplace Operator. $\Delta^2$ can be split into four sub-matrices:

$$\Delta^2 = (\Delta^2)_{bb}(\Delta^2)_{bu}(\Delta^2)_{ub}(\Delta^2)_{uu} \tag{3.4}$$

It's obvious to see $-(\Delta^2)_{uu}^{-1}(\Delta^2)_{ub}$ is the interpolation operator $P_u$ for $g(u)$ so that:

$$g(u) = -(\Delta^2)_{uu}^{-1}(\Delta^2)_{ub}g(b) = P_u g(b) \tag{3.5}$$

The Figure 3.4 is a one dimensional demonstration for BHA. BHA is an interpolation of irregularly spaced two-dimensional data points. In total, BHA can be applied in 3 ways in our arrangement model: interpolating geodesic distance matrix, interpolating spring network energy and a mixed model containing both.

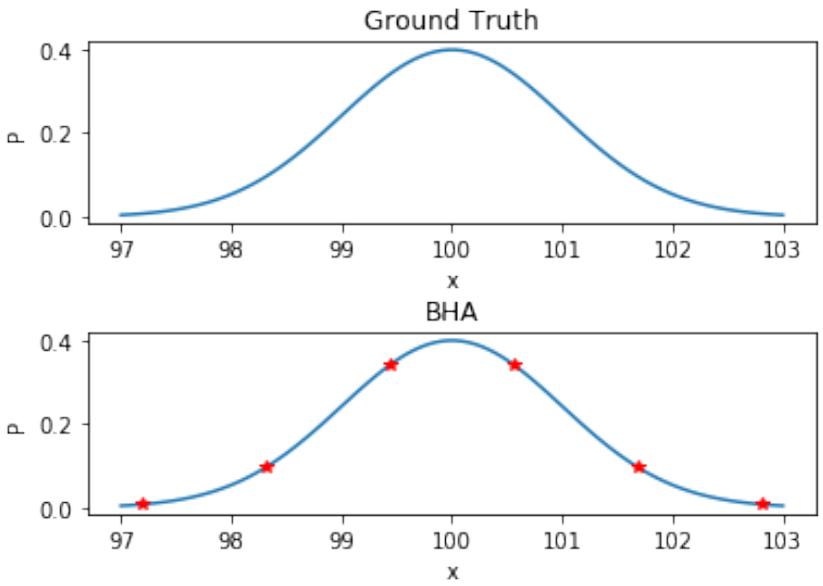


Figure 3.4: One dimensional comparison between ground truth and BHA. x axis is a sample location and y axis is the probability. BHA draw some landmarks from the ground truth and interpolate the whole probability distribution. The interpolated distribution might differ from the ground truth a little bit

### 3.4.1 BHA1

BHA1 interpolates the full geodesic distance matrix $\mathcal{D}_n$ from a smaller geodesic distance matrix $\mathcal{D}_l$. In our PrAGMATiC model, there are three steps in BHA1:

1. We choose $l$ landmarks. The first landmark is chosen at random. Each subsequent landmark is then chosen as the point with the largest minimum distance to any of the current landmarks, until $l$ landmarks have been selected. These landmarks are considered as known points in BHA. Compute the geodesic distance matrix between the $l$ selected landmarks as the landmarks distance matrix $\mathcal{D}_l$. $\mathcal{D}_l$ has the dimension of $l \times l$ .

2. Since we are interpolating to not only the unknown points but the full surface mesh, we compute the interpolation operator $P$ which has the dimension of

$n \times l$ by vertical joining identity matrix and $P_u$ :

$$P = P_n = I_l P_u = I_l - (\Delta^2)_{uu}^{-1}(\Delta^2)_{ub} \tag{3.6}$$

where $I_l$ is the identity matrix in dimension of $l \times l$

3. Interpolate the full geodesic distance matrix $\mathcal{D}_n$ with $P$ and $\mathcal{D}_l$ via BHA:

$$\mathcal{D}_n^{BHA} = P\mathcal{D}_l P^T \tag{3.7}$$

Here, BHA can be thought of as performing two interpolation operations. First, the product $P\mathcal{D}_l$ interpolates geodesic distances from the landmarks to all the points, approximating the $n \times l$ submatrix formed by the columns of $P\mathcal{D}_l$ corresponding to the landmarks. Right-multiplying by $P^T$ then interpolates each row of $P\mathcal{D}_l$ across the entire surface mesh, giving the full $n \times n$ approximation $\mathcal{D}_n^{BHA}$. Notice the number of landmarks is much smaller than the number of sparse points. Given 150k points in our cortex map, we take 3k of them, about 2% of the total, as landmarks. $\mathcal{D}_l$ has a small dimension and can be precached. The number of landmarks is the trade-off for BHA's efficiency and accuracy. The less landmarks we have, the faster BHA interpolates but less accurately.

### 3.4.2 BHA2

In the Gibbs sampler, when we update the location of centroid $i$, we have to calculate the spring energy between the other fixed centroids $H_{\setminus i}$ and a candidate point $l^*$ scanning all over the surface mesh, according to equation 2.5 and 2.6 resulting in Figure 2.3. Instead of searching the whole surface mesh, we can just search the landmarks we found in BHA1 as candidate. Then we apply BHA to interpolate the spring energy from landmarks to all the points as new candidates. The full energy for the spring network $E_l$ has a dimension of $1 \times n$ and energy of landmarks for

the spring network $E_l$ has a dimension of $1 \times l$. BHA2 interpolates energy of the spring network by right-multiplying by $P^T$ then interpolates each row of $E_l$ across the entire surface mesh:

$$E_n^{BHA} = E_l P^T \tag{3.8}$$

where $P^T$ is the transpose of interpolation operator $P$. Since interpolations of both BHA1 and BHA2 are from the same landmarks to the same surface mesh, so the $P$ stays the same in BHA1 and BHA2.

### 3.4.3 BHA3

BHA3 combines BHA1 and BHA2, interpolating both the geodesic distance matrix and the energy of the spring network via Equation 3.7 and Equation 3.8.

# Chapter 4

# Approximation comparison

The previous chapter gives us several approximation methods. Which one is the best? To determine this, we generated samples under each approximation method and then compared them to the ground truth. We use time, jumping distance, Entropy (instability) and Kullback–Leibler divergence as our metrics for the comparison. We take 10 $\beta$ points exponentially ranging from $1.0 \times 10^{-5}$ to $8.1 \times 10^{-2}$ to run the arrangement model for the ground truth and 6 approximation methods. Notice that both the temperature $T$ or the inverse temperature $\beta$ are defined by us and dimensionless in the model. Number of centroids (tiles), $N_c$, is 128 and Number of iterations (sweeps), $N_i$, is 200. Close and sparse is labeled as "$c\&s$" and ground truth as "all" in the comparison figures. Figure 4.1 shows the final configuration of tiles after 200 iterations of sampling at $\beta$=$8.1 \times 10^{-2}$ for ground truth model.

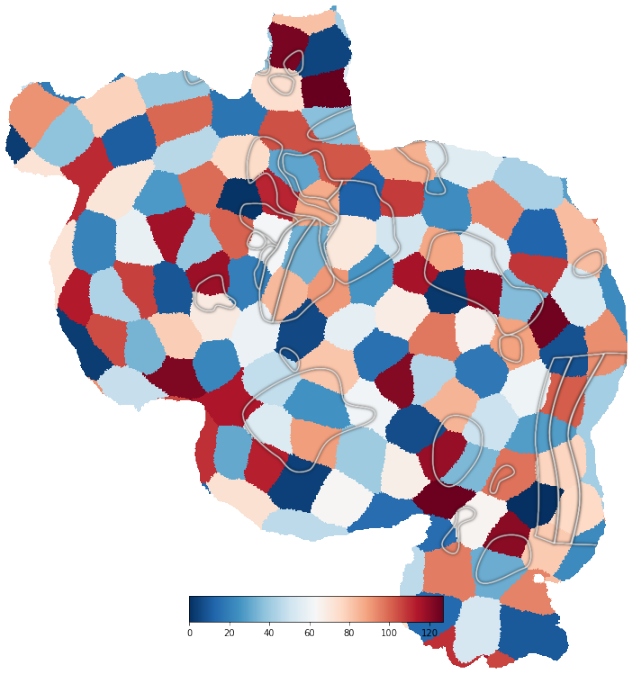


Figure 4.1: The final configuration of cortical tiles after 200 iterations of sampling at $\beta$=8.1 $\times$ $10^{-2}$ for ground truth model. There are $N_c = 128$ tiles in total.

## 4.1 Time

Time is the most important metric we use for the modeling efficiency comparison. We average the time over iterations and plot the time vs $\beta$ for the ground truth and approximation models in Figure 4.2. We can see the three methods "all, BHA2 and close" are the most expensive at the level of $4 \times 10^3$ seconds. BHA1 and BHA3 are faster around the level of $8 \times 10^2$ seconds due to interpolation of geodesic distance. Sparse and $c\&s$(short for close and sparse) are very fast between $10^1$ and $10^2$ seconds for different $\beta$ because the geodesic distance matrices are precached. Sparse method's time is pretty constant while $c\&s$ method's time decreases as $\beta$ increase. Overall, when $\beta < 5 \times 10^{-4}$, the sparse is fastest and when $\beta > 5 \times 10^{-4}$ , the $c\&s$ is fastest.

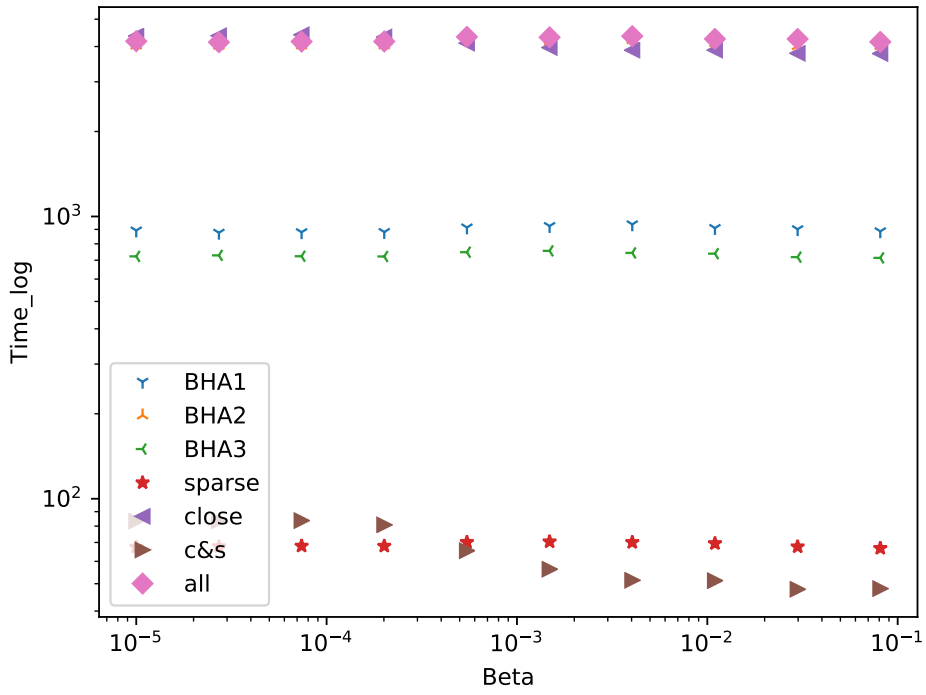


Figure 4.2: Time(s) vs. $\beta$. As As $\beta$ increases, the time of $c\&s$ and close decrease. Other methods takes relative constant time no matter how $\beta$ changes. The all, BHA2 and close are most expensive; BHA1 and BHA3 are in the middle level; sparse and $c\&s$ are most efficient. When $\beta < 5 \times 10^{-4}$, the sparse is faster and when $\beta > 5 \times 10^{-4}$ , the $c\&s$ are faster.

## 4.2 Jumping distance

Jumping distance is the change of location of each centroid at each iteration measured in geodesic distance defined in Euqation 3.1. In our comparison, we use the average of jumping distance over all centroids and all iterations as one of our metrics of the dynamics of the system. Besides, jumping distance is also used as the

threshold value for close area approximation.

$$JD = \frac{1}{N_t}\frac{1}{N_C}\sum_i\sum_t \mathcal{D}(C_i^t, C_i^{t+1}) \tag{4.1}$$

Figure 4.3 is the loglog plot of jumping distance(mm) vs. $\beta$. The jumping distances decreases when $\beta$ increases. All the approximation matches with the ground truth pretty well until the $\beta$ gets high around $2 \times 10^{-2}$. When $\beta$=$8.1 \times 10^{-2}$, the system of close approximation is almost frozen with jumping distance almost zero. Frozen state means the sampler gets stuck in a local minima and not globally optimal. "BHA1 and BHA3" are more dynamic than "ground truth and BHA2". "$c\&s$" lies between the ground truth and close.

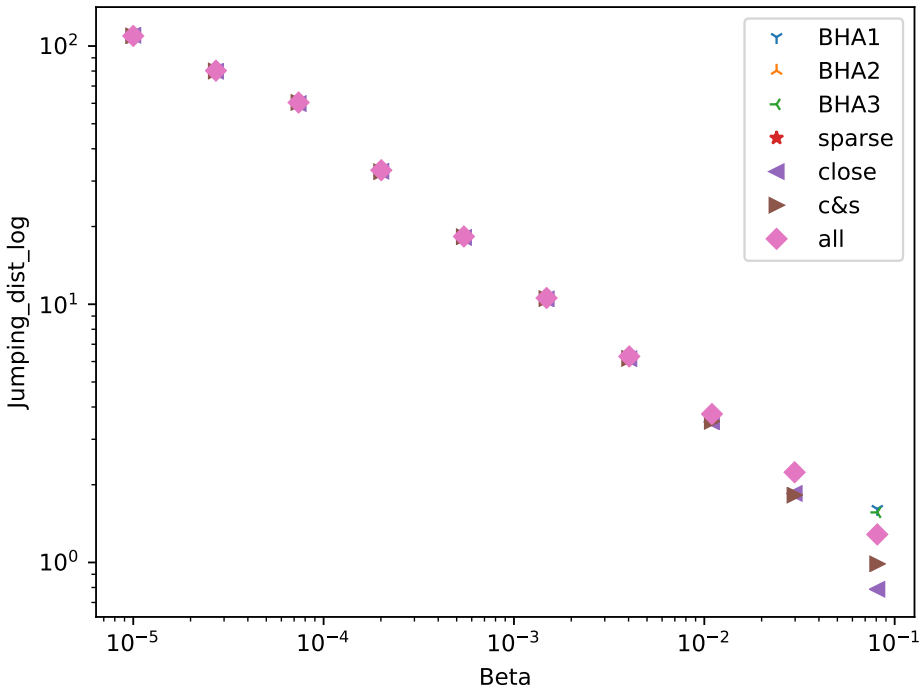


Figure 4.3: Jumping distance(mm) vs. $\beta$. As $\beta$ increases, temperature decreases, the system cools down and the jumping distances decreases. All the approximation matches very well unless very high $\beta$. At high $\beta$, the BHA overestimate the jumping distance while sparse, close and $c\&s$ underestimate the jumping distance

## 4.3 Accuracy

### 4.3.1 Voronoi table

At every iteration, all the points on the surface mesh can find its nearest centroid and therefore tile the cortical surface. To compare the accuracy among different sampling methods, we need to keep track of the sampling (tiling) results. We construct a Voronoi table to track every point's tile assignment over iterations as a measure of sampling. The Voronoi table is initialized as a $n_C \times n$ zero matrix, where $n_C$ is

number of centroids and $n$ is the number of total points. At one iteration, the point $l$ finds its nearest centroid $C_i$; we then add the counting integer one to the cell $(i, l)$ of Voronoi table. Voronoi table is an accumulated counter of the points belonging to which centroids (tiles). It's easy to change the counter to a probability distribution through dividing by the number of iterations $N_i$.

### 4.3.2 Entropy (instability)

From the Voronoi table, we can get the distribution or probability distribution of tile belonging of every point on the surface mesh. How to visualize the Voronoi table on the cortical surface? One intuition is to calculate the entropy for every point. Specifically, entropy is a logarithmic measure of the number of states with significant probability of being occupied. The definition of the entropy is expressed in terms of a discrete set of probabilities $p_i$ so that

$$\mathcal{E} = -\sum_{i=1}^{N_c} p(x_i) \log p(x_i) \tag{4.2}$$

The summation is over all the centroids of the system, and $p_i$ is the probability that point is in the $i$th centroid. For every point on the surface mesh, we can take its corresponding column of the voronoi table and convert it to probability distribution. Then we can calculate its entropy. Generally speaking, entropy is a measure of the number of ways in which a system may be arranged, often taken to be a measure of disorder (the higher the entropy, the more the disorder).[10] Here, the entropy is a interpreted as instability. The higher entropy is, the more unstable the point behaves during the sampling by changing its belonging centroid (tile) at a larger probability. Figure 4.4 shows cortical entropy map after 200 iterations of sampling at $\beta = 8.1 \times 10^{-2}$ for ground truth model. Points inner inside the tile has relative lower entropy since there is less chance for them to change tile assignment while the points near boundary are the reverse.

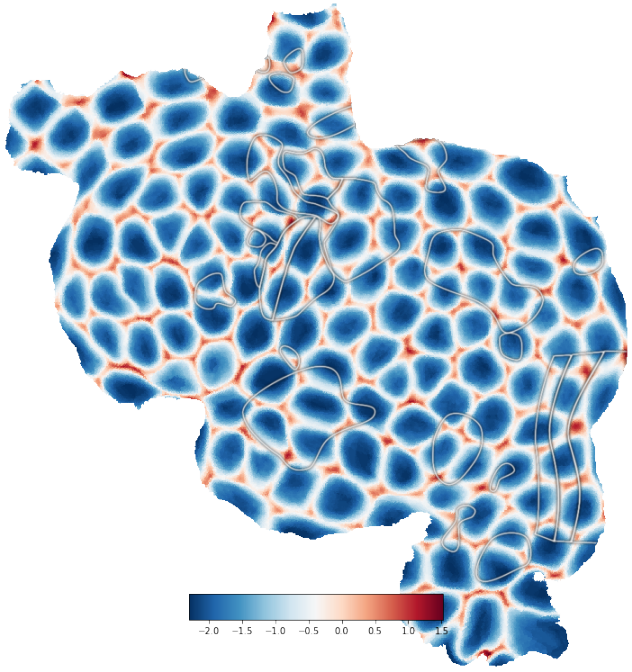


Figure 4.4: The cortical entropy(instability) map after 200 iterations of sampling at $\beta$=$8.1 \times 10^{-2}$ for ground truth model. Blue means low entropy and red means high entropy. Points inside the tile has relative lower entropy since there is less chance for them to change tile assignment while the boundary points are the reverse.

If we averaged the instability across locations on the cortical surface, we can get instability for every method at every $\beta$. The results are plotted in Figure 4.5. As $\beta$ increases, temperature decreases, the system cools down and the instability decreases. All the approximation matches very well unless $\beta$ is very high. At $\beta$=$8.1 \times 10^{-2}$, the *cs* overestimate the instability while other approximations close to ground truth.

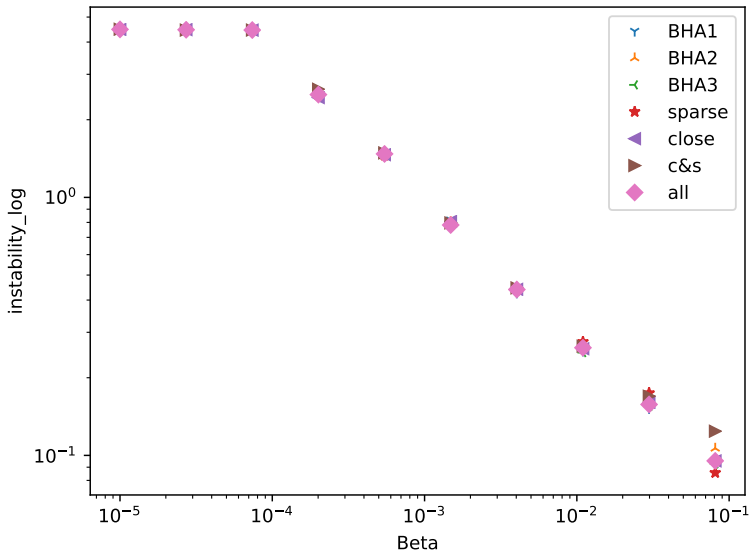


Figure 4.5: Instability vs. $\beta$. As $\beta$ increases, temperature decreases, the system cools down and the instability decreases. All the approximation matches very well unless very high $\beta$. At $\beta$=$8.1 \times 10^{-2}$, the *cs* overestimate the instability while other approximations close to ground truth

### 4.3.3 Kullback–Leibler divergence

Entropy enablse us to visualize the Voronoi table on the cortical surface. But how to compare the Voronoi table among different methods? The Kullback–Leibler divergence (KL divergence) measures how different one probability distribution is from a reference probability distribution. The higher KL divergence is, the more the two probability distributions differ from each other. The minimum KL divergence is 0 indicating that the two distributions compared are identical.[20] For discrete probability distributions $P_1$ and $P_2$ defined on the same probability space, the KL divergence of $P_1$ from $P_2$:

$$D_{KL}(P_1 \parallel P_2) = -\sum_{x \in \mathcal{X}} P_1(x) \log \left( \frac{P_2(x)}{P_1(x)} \right) \tag{4.3}$$

In our comparison, $P_1$ is the probability distribution from $l$th column of the approximation methods' Voronoi table for the $l$th location; $P_2$ is the probability distribution from $l$th column of the ground truth' Voronoi table for the $l$th location. We then averaged the KL divergence averaged across locations on the cortical surface. The KL divergence will be used as the metric for accuracy in comparison. The lower KL divergence, the more accurate the approximation method is. Figure 4.6 shows the KL divergence vs. $\beta$ in loglog plot. When $\beta < 10^{-4}$, the temperature is so high that all approximations are equally bad with high KL divergence values. When $10^{-4} < \beta < 10^{-2}$, all approximations are have higher accuracy and lower KL divergence values as $\beta$ increases; there is not too much difference among those approximation methods. When $\beta > 10^{-2}$, close and BHA2 keep doing better and close approximation is the ultimate winner; BHA1 and BHA3 are staying at the same KL divergence level; sparse and $c\&s$ divergence values increase as $\beta$ increases which means they are doing bad.

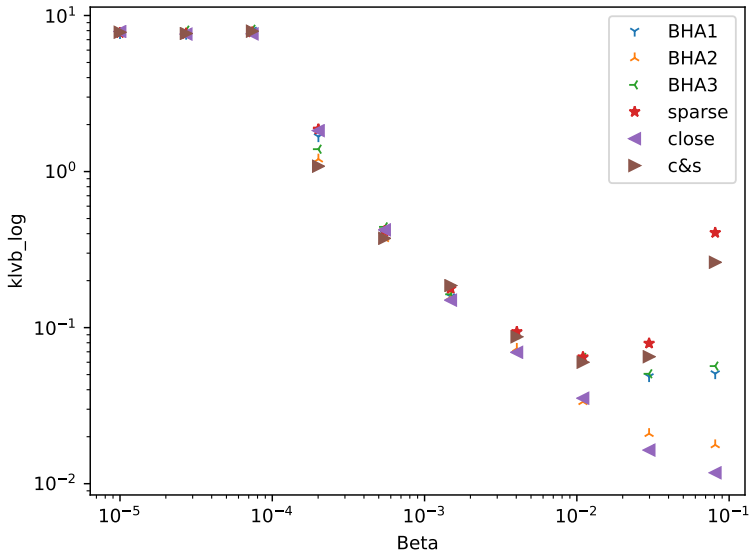


Figure 4.6: KL divergence vs. $\beta$. When $\beta < 10^{-4}$, the temperature is so high that all approximations are equally bad with high KL divergence values. When $10^{-4} < \beta < 10^{-2}$, all approximations are have higher accuracy and lower KL divergence values as $\beta$ increases; there is not too much difference among those approximation methods. When $\beta > 10^{-2}$, close and BHA2 are keeping doing better and close approximation is the ultimate winner; BHA1 and BHA3 are staying at the same KL divergence level; sparse and $c\&s$ divergence values increase as $\beta$ increases which means they are doing bad.

We should mention that KL divergence among the Voronoi tables also depends on the number of iterations we run the sampler. The more iterations we run, the lower KL divergence we get and the less bias the KL divergence will be. In other words, our KL divergence values are biased due to the limitation of iterations. However, all the approximation and ground truth have run the same iterations. All the KL divergence we calculated have the same bias so the comparison relationships still hold. Since the KL divergence is the most important metric for accuracy, we selected three approximation methods with relatively good performance in time and accuracy (BHA1, close, $cs$) and run for 1000 iterations. The results are plotted in

Figure 4.7 and they are consistent with 4.6. When $\beta < 2 \times 10^{-3}$, three approximations have similar KL divergence values. When $2 \times 10^{-3} < \beta < 3 \times 10^{-2}$, close approximation has lower KL divergence; BHA1 and *cs* approximations have relative high values. there is not too much difference among those approximation methods.When $\beta > 3 \times 10^{-2}$, close is keeping doing better and close approximation is the ultimate winner; BHA1 stays in the middle and *c&s* goes up.

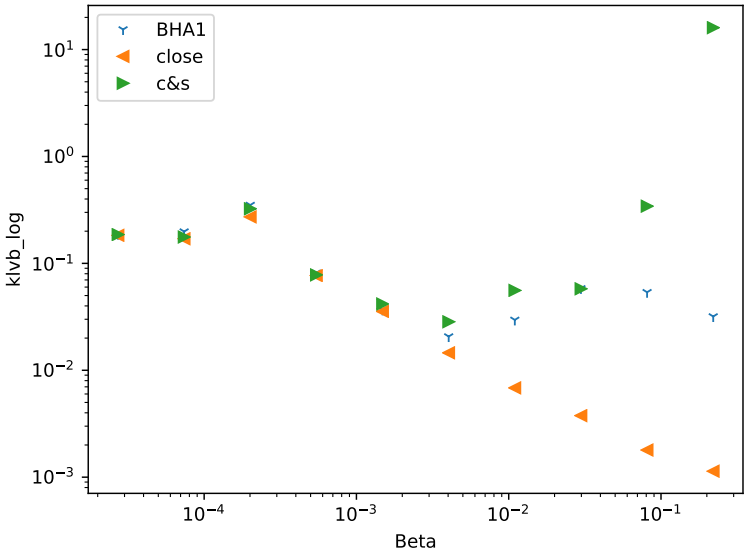


Figure 4.7: KL divergence of (BHA1, close, *c&s*) from the ground truth vs. $\beta$. $\beta$ ranges from $2.7 \times 10^{-5}$ to $2.2 \times 10^{-1}$. When $\beta < 2 \times 10^{-3}$, three approximations have similar KL divergence values. When $2 \times 10^{-3} < \beta < 3 \times 10^{-2}$, close approximation has lower KL divergence; BHA1 and *c&s* approximations have relative high values. there is not too much difference among those approximation methods. When $\beta > 3 \times 10^{-2}$, close is keeping doing better and close approximation is the ultimate winner; BHA1 stays in the middle and *c&s* goes up.

# Chapter 5

# Conclusion

In order to register different cortical surfaces, we have developed an energy-based model of areas tiling human cerebral cortex across individuals. We applied several approximation methods including biharmonic matrix approximation(BHA), sparsity, close area, and $c\&s$(mixed of close and sparse) for fast computation of the arrangement model. By comparing efficiency in time and storage and accuracy in Kullback–Leibler divergence among them we find out the best strategy below to make the model practical. From the perspective of time, when $\beta < 5 \times 10^{-4}$, the sparse is fastest and when $\beta > 5 \times 10^{-4}$ , the $c\&s$ is fastest. BHA1 and BHA3 are in the middle but others are too expensive. From the perspective accuracy, when $\beta < 5 \times 10^{-3}$, all approximations have similar accuracy and when $5 \times 10^{-3} < \beta < 10^{-2}$, both BHA2 and close are equally the most accurate; when $\beta > 10^{-2}$ close is the most accurate and BHA2 is following closely; BHA1 and BHA3 are in the middle. From the perspective of storage space, all the three BHA methods take smaller space than sparse and $c\&s$. The close and ground truth's geodesic distance matrix are too large to store in the memory. Thus our conclusion is that when running the model by increasing $\beta$ exponentially from $10^{-5}$ to $10^{-1}$, we should choose different approximation methods at different $\beta$ regions:

1. when $10^{-5} < \beta < 5 \times 10^{-4}$, choose sparse approximation

2. when $5 \times 10^{-4} < \beta < 5 \times 10^{-3}$, choose $c\&s$ approximation

3. when $5 \times 10^{-3} < \beta < 10^{-1}$, BHA1, BHA3 and close approximation are all acceptable. The close is more accurate but BHA1 and BHA3 are faster with less accuracy.

Among the three BHA methods,

1. BHA1, interpolating the geodesic distance, saves time with middle level of accuracy.

2. BHA2, interpolating energy of the spring network, does not save time but has high accuracy which is similar to the close area approximation. However, the close area approximation outperforms BHA2 in accuracy and storage with the same amount of time taken. BHA2 should be abandoned.

3. BHA3, mix of BHA1 and BHA2, has similar results as BHA1. This should be credited to BHA1 since BHA1 and BHA2 are quite independent. BHA3 should be abandoned as well since it contains the BHA2 which is outperformed by close area approximation.

We should take the BHA1 which interpolates the geodesic distance, as the only BHA in our model. In conclusion, when the system is hot, $\beta$ being low, the sparse and $c\&s$ approximation is the most practical method. When the system cools down, $\beta$ being high, the BHA is the most practical method.

# Bibliography


[1] James S Gao, Alexander G Huth, Mark D Lescroart, and Jack L Gallant. Pycortex: an interactive surface visualizer for fmri. *Frontiers in neuroinformatics*, 9:23, 2015. URL: `https://www.frontiersin.org/articles/10.3389/fninf.2015.00023/full`.

[2] Ilker Yildirim. Bayesian inference: Gibbs sampling. *Technical Note, University of Rochester*, 2012. URL: `http://www.mit.edu/~ilkery/papers/GibbsSampling.pdf`.

[3] Alexander G Huth, Thomas L Griffiths, Frederic E Theunissen, and Jack L Gallant. Pragmatic: A probabilistic and generative model of areas tiling the cortex. *arXiv preprint arXiv:1504.03622*, 2015. URL: `https://arxiv.org/abs/1504.03622`.

[4] Kenneth S Saladin. *Human anatomy*. McGraw-Hill, 2011.

[5] Roberto Toro, Michel Perron, Bruce Pike, Louis Richer, Suzanne Veillette, Zdenka Pausova, and Tomáš Paus. Brain size and folding of the human cerebral cortex. *Cerebral cortex*, 18(10):2352–2357, 2008.

[6] Dean V Buonomano and Michael M Merzenich. Cortical plasticity: from synapses to maps. *Annual review of neuroscience*, 21(1):149–186, 1998.

[7] Bruce Fischl, Martin I Sereno, Roger BH Tootell, and Anders M Dale. High-resolution intersubject averaging and a coordinate system for the cortical surface. *Human brain mapping*, 8(4):272–284, 1999. URL: https://doi.org/10.1002/(SICI)1097-0193(1999)8:4<272::AID-HBM10>3.0.CO;2-4.

[8] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006. URL: http://yann.lecun.com/exdb/publis/orig/lecun-06.pdf.

[9] Junbo Zhao, Michael Mathieu, and Yann LeCun. Energy-based generative adversarial network. *arXiv preprint arXiv:1609.03126*, 2016.

[10] LD Landau and EM Lifshitz. Statistical physics (course of theoretical physics vol 5). 1958.

[11] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

[12] Alexander G. Huth, Wendy A. de Heer, Thomas L. Griffiths, Frederic E. Theunissen, and Jack L. Gallant. Natural speech reveals the semantic maps that tile human cerebral cortex. *Nature*, page 453–458, April 2016. URL: https://www.nature.com/articles/nature17637.

[13] Harold J Larson and Harold J Larson. *Introduction to probability theory and statistical inference*, volume 12. Wiley New York, 1969.

[14] Neal Noah Madras. *Lectures on monte carlo methods*, volume 16. American Mathematical Soc., 2002.

[15] Scott M Lynch. *Introduction to applied Bayesian statistics and estimation for social scientists*. Springer Science & Business Media, 2007.

[16] Paul A Gagniuc. *Markov chains: from theory to implementation and experimentation*. John Wiley & Sons, 2017.

[17] Richard J Rossi. *Mathematical Statistics: An Introduction to Likelihood Based Inference*. John Wiley & Sons, 2018.

[18] Javier S Turek and Alexander G Huth. Efficient, sparse representation of manifold distance matrices for classical scaling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2850–2858, 2018.

[19] Keenan Crane, Clarisse Weischedel, and Max Wardetzky. Geodesics in heat: A new approach to computing distance based on heat flow. *ACM Transactions on Graphics (TOG)*, 32(5):152, 2013. URL: `https://dl.acm.org/citation.cfm?id=2516977`.

[20] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.